Table of Contents

As already mentioned under Connections, objects are serialized during the transmission between client and enterprise tier. However, instead of the standard Java serialization, special technology-independent serialization mechanisms are used.

The so-called serializers have to fulfill the ISerializer interface.

Why Specific Serializers?

The communication between Java and C#, Flex, etc. would hardly be possible using standard Java serialization or the Java serialization would have to be taken over. The effort would be enormous.

Existing serializing frameworks, such as Hessian, do not support all necessary objects, such as BigDecimal.

For this reason, a separate serializer was implemented, the UniversalSerializer.

However, using the definition of ISerializer, frameworks such as Hessian can be integrated in JVx.

Example

We want to implement a serializer that uses the standard Java serialization.

JavaSerializer.java

```java
public class JavaSerializer implements ISerializer
{
   public Object read(DataInputStream in) throws Exception
   {
      ObjectInputStream ois = new ObjectInputStream(in);

      return ois.readObject();
   }

   public void write(DataOutputStream out, Object object) throws
Exception
   {
      ObjectOutputStream oos = new ObjectOutputStream(out);

      oos.writeObject(object);
   }
}
```